Rattrapage Challenge S2 - 5IW

Matières

Matières
Golang
Flutter

Présentation du projet

Le but de ce projet est de réaliser un SaaS pour la gestion de kermesses (fêtes de fin d'année) dans les écoles primaires. Vous devrez développer une application mobile **Flutter** ainsi qu'une API **Golang** pour permettre les fonctionnalités décrites ci-après.

Le projet est à réaliser **SEUL**.

Le SaaS doit permettre la gestion de plusieurs kermesses mais il n'est pas demandé de gérer des écoles. Les participants peuvent interagir avec les stands via un système de jetons. Ces jetons sont achetables avec de l'argent réel par les parents d'élèves et ils peuvent en attribuer une partie à leurs enfants.

Les jetons servent également à acheter des tickets pour une tombola globale avec des lots et gérée par le ou les organisateurs de la kermesse.

Fonctionnalités attendues

- Gestion des participants
  - Élève (enfant)
    - Peut interagir avec un stand via des jetons
    - Peut acheter des billets de Tombola
  - Parent

        - Lié à un ou plusieurs élèves
        - Peut interagir avec un stand via des jetons
        - Peut visualiser les interactions de ses enfants
        - Peut acheter des jetons
        - Peut distribuer des jetons à ses enfants
    - Teneur de stand
        - Peut gérer les stocks (nourriture / boisson)
        - Peut faire payer les consommations ou activités en jeton
        - Peut attribuer des points aux participants des activités
    - Organisateur
        - Peut visualiser tous les stands de la kermesse
            - Stock, jetons dépensés, points attribués
        - Peut visualiser les recettes globales
            - Jetons achetés
        - Peut chater avec les teneurs de stand pour gérer les problèmes
        - Peut gérer les lots de la tombola
        - Peut faire le tirage de la tombola par rapport aux tickets achetés
        - Peut visualiser le classement des points des activités pour attribuer des lots supplémentaires
- Gestion des kermesses
    - Liée à un ou plusieurs organisateurs
    - Liée à des participants
    - Liée à des stands
    - Visualisation d’un plan interactif de la kermesse avec les stands
    - Gestion d’une tombola
- Gestions des stands
    - Types : vente de nourriture / vente de boisson / activité rapportant des points

    - Gestion des stocks
    - Gestion des jetons
    - Attribution des points (activité seulement)

Fonctionnalités techniques

- **Golang**
    - API REST
        - Système d'authentification (au minimum via login / mot de passe)
        - Gestion multi-rôles (utilisateur simple, administrateur...), vérification des droits et filtrage des résultats lors des requêtes en fonction du rôle de l'utilisateur connecté
        - Interface Swagger UI pour interagir avec l'API
        - API CRUD sur les éléments fonctionnels de l'application
    - API Web Socket fournissant au moins une fonctionnalité pseudo temps-réel
    - Intégration d'un système de paiement
    - Configuration modifiable (via fichier de conf, variables d'environnement, base de données...)
    - Base de données relationnelle avec jeu de données conséquent représentatif des cas d'usages de l'application
- **Flutter**
    - Application mobile (iOS & Android)
        - Écrans différents en fonction des rôles de l'utilisateur
        - Envoi de notifications push
        - Utilisation d'une technologie pseudo temps réel
        - Intégration d'un système de paiement

Contraintes

- **Golang**

- Utilisation de Go 1.22.1
- Base de données PostgreSQL
- API stateless, avec pagination des résultats
- Configuration modifiable (fichier de conf, variables d'environnement, base de données...)
- Erreurs :
  - pas de "panic" dans le code, les erreurs doivent être correctement gérées
  - renvoyer des erreurs fonctionnelles par l'API et afficher le details des erreurs techniques dans les logs
- Sécurité :
  - API hébergée en HTTPS avec certificat SSL
  - Respect des normes de développement standard (OWASP...)
- Clean code :
  - Le code doit être découpé en fichiers et dossiers cohérents et respecter les règles du clean code et SOLID (découpage, noms des variables et fonctions...)
  - Code formaté et indenté avec Gofmt

- **Flutter**
  - Utilisation de plusieurs émulateurs ou terminaux pendant la soutenance pour faire les démonstrations nécessaires en fonction des différents profils
  - L'application devra être visuellement responsive en fonction de la taille du téléphone / navigateur
  - Le code rendu doit être propre et indenté correctement via les outils de base mis à disposition

Rendu

- **Dépôt(s) GitHub**
  - **⚠ Les commits doivent être signés via une clef GPG**

    - `README.md`
        - NOM Prénom de l'étudiant avec compte GitHub associé
        - Toute information semblant nécessaire...
- **Mise en production**
    - Pour valider, votre projet doit être impérativement mis en production pour le jour de la soutenance et ce durant au moins 2 semaines. Vous pouvez utiliser un VPS ou des services comme Heroku, Firebase, etc.
- **Golang**
    - Dépôt Git contenant le code du backend
    - Fichier `README.md` détaillant
        - La listes des binaires et les fonctionnalités fournies
        - Les procédure de compilation et d'exécution du ou des binaires
        - Les procédures de lancement des tests unitaires
        - Un exemple de configuration pour lancement sur poste de travail local
    - Archive au format **ZIP sur MyGES**
        - Eléments tiers nécessaires (configuration, certificats, variables d'environnement...)
- **Flutter**
    - Dépôt Git contenant le code de l'application Flutter
    - Archive au format **ZIP sur MyGES**
        - Éléments tiers nécessaires (certificats, variables d'environnement...)
        - Application Android de production sous forme d'APK signé et installable

Organisation de la soutenance

Un ordre de passage sera défini pour soutenir votre rattrapage. Vous aurez 30 minutes, à répartir selon le schéma suivant :

- 20 minutes de présentation / démonstration

- 10 minutes de questions / réponses

Pour les 20 premières minutes, vous pouvez suivre la ligne directrice suivante :

- Présentation rapide du concept et de l’équipe (à l’oral, 30 secondes)
- Présentation des grandes fonctionnalités du projet (2 minutes max.)
- Démonstration de votre projet à travers les différents parcours utilisateurs